```
/*************************************************************************
                           Constante  -  description
                             -------------------
    début                : mer jan 31 2007
    copyright            : (C) 2007 par IF3134 - NGUYEN Hoang Thanh
    e-mail               :
*************************************************************************/

//---------- Interface de la classe <Constante> (fichier Constante.h) ------
#if ! defined ( CONSTANTE_H )
#define CONSTANTE_H

//--------------------------------------------------- Interfaces utilisées
#include <string>
#include "Expression.h"
using namespace std;
//------------------------------------------------------------- Constantes

//------------------------------------------------------------------ Types

//------------------------------------------------------------------------
// Rôle de la classe <Constante>
//      Représenter une expression Constante
//      Fils de la classe Expression
//
//------------------------------------------------------------------------

class Constante : public Expression
{
//----------------------------------------------------------------- PUBLIC

public:
//----------------------------------------------------- Méthodes publiques
    virtual void AffichageInfixe( );
    // Mode d'emploi :
    //      Afficher sous forme infixé de la Constante
    //
    // Contrat : Aucun
    //

    virtual void Afficher ( );
    // Mode d'emploi :
    //      Afficher tous les attributs. La valeur de type doit être 0.
    //
    // Contrat : Aucun
    //

    virtual Expression * Simplifier( );
    // Mode d'emploi :
    //      Créer la nouvelle constante égale à elle-même
    //
    // Contrat : Aucun
    //

    virtual bool EstConstante(int uneConstante) const;
    // Mode d'emploi :
    //      <uneConstante> : La valeur à comparer avec l'instance
    //      Renvoyer true si la valeur de la constante égale à
    //      uneConstante. Renvoyer false dans le cas contraire
    //
    // Contrat : Aucun
    //

    virtual bool Sauver(ofstream & ficDest);
    // Mode d'emploi :
    //      <ficDest> : ofstream lié au fichier de sauvegarde
    //      Sauver l'instance de la classe dans le fichier lié à ficDest.
    //      Retourner TRUE si la sauvegarde est effectuée, FALSE dans le
```

```
    //      cas contraire
    //
    // Contrat : Aucun
    //

//------------------------------------------------- Surcharge d'opérateurs
    Constante & operator = ( const Constante & unConstante );
    // Mode d'emploi :
    //      Méthode non réalisé (appel interdit)
    //
    // Contrat :
    //

//-------------------------------------------- Constructeurs - destructeur
    Constante ( const Constante & unConstante );
    // Mode d'emploi (constructeur de copie) :
    //      Méthode non réalisé (appel interdit)
    //
    // Contrat :
    //

    Constante (int uneValeur);
    // Mode d'emploi :
    //      <uneValeur> : la valeur que va recevoir la Constante
    //      Crée une Constante à partir du entier uneValeur
    //
    // Contrat :
    //      uneValeur doit être soit 0, soit 1.

    virtual ~Constante ( );
    // Mode d'emploi :
    //      Destructeur de l'objet Constante
    //
    // Contrat :
    //      Aucun

//------------------------------------------------------------------ PRIVE

protected:
//----------------------------------------------------- Méthodes protégées

//----------------------------------------------------- Attributs protégés
    int valeur;
};

//-------------------------- Autres définitions dépendantes de <Constante>

#endif // CONSTANTE_H
```